Computation of the Probability of Initial Substring Generation by Stochastic Context-Free Grammars

Frederick Jelinek*
John D. Lafferty*
IBM T. J. Watson Research Center

Speech recognition language models are based on probabilities $P(W_{k+1} = v \mid w_1 w_2, \ldots, w_k)$ *that the next word* W_{k+1} *will be any particular word* v *of the vocabulary, given that the word sequence* $w_1, w_2, \ldots, w_k$ *is hypothesized to have been uttered in the past. If probabilistic context-free grammars are to be used as the basis of the language model, it will be necessary to compute the probability that successive application of the grammar rewrite rules (beginning with the sentence start symbol s) produces a word string whose initial substring is an arbitrary sequence* $w_1, w_2, \ldots, w_{k+1}$. *In this paper we describe a new algorithm that achieves the required computation in at most a constant times* k^3*-steps.*

1. Introduction

The purpose of this article is to develop an algorithm for computing the probability that a stochastic context-free grammar (SCFG) (that is, a grammar whose production rules have attached to them a probability of being used) generates an arbitrary initial substring of terminals. Thus, we treat the same problem recently considered by Wright and Wrigley (1989) from the point of view of LR grammars.

Probabilistic methods have been shown most effective in automatic speech recognition. Recognition (actually transcription) of natural unrestricted speech requires a "language model" that attaches probabilities to the production of all possible strings of words (Bahl et al. 1983). Consequently, if we believe that word generation can be modeled by context-free grammars, and if we want to base speech recognition (or handwriting recognition, optical character recogition, etc.) on such models, then it will become necessary to embed them into a probabilistic framework.

In speech recognition we are presented with words one at a time, in sequence, and so we would like to calculate the probability $P(s \rightarrow w_1 w_2 \ldots w_k \ldots)$ that an arbitrary string $w_1 w_2 \ldots w_k$ is the initial substring of a sentence generated by the given SCFG.[1]

* P.O. Box 218, Yorktown Heights, NY 10598

1 In fact, in speech recognition (Bahl et al. 1983) we are presented with a hypothesized past text (the history) $w_1 w_2 \ldots w_k$ and are interested in computing, for any arbitrary word v, the conditioned probability $P(W_{k+1} = v \mid w_1 w_2 \ldots w_k)$ that the next word uttered will be v given the hypothesized past $w_1 w_2 \ldots w_k$. Assuming that successive sentences s are independent of each other (a rather dubious assumption justifiable only by a lack of adequate understanding of how one sentence influences another), we may as well take the view that w_1 is the first word of the current sentence and that w_k is not the last. Then

$$P(W_{k+1} = v \mid w_1 w_2 \ldots w_k) = \frac{P(s \rightarrow w_1 w_2 \ldots w_k v \ldots)}{P(s \rightarrow w_1 w_2 \ldots w_k \ldots)}$$

Hence our interest in the calculation of $P(s \rightarrow w_1 w_2 \ldots w_k \ldots)$.

2. Definition of Stochastic Context-Free Grammars

We will now define stochastic context free grammars (SCFGs) and establish some notation. We will use script symbols for sets, lowercase letters for elements of the sets or specific string items, and capitals for variables. We start with a vocabulary $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ whose elements, words v_i, are the terminal symbols of the language. We next list a set of nonterminals $\mathcal{G} = \{g_1 = s, g_2, \ldots, g_M\}$ whose elements g_j are grammatical phrase markers. They include the distinguished phrase marker s, the sentence "start" symbol. The purpose of our grammar is to generate sentences $w_1 w_2 \ldots w_n$ of varying length n. The generation is accomplished by use of production rules, belonging to a set $\mathcal{R}$, that rewrite individual phrase markers as sequences of phrase markers or words. For simplicity of manipulation but without loss of generality, we will limit the productions to the Chomsky Normal Form (CNF). That is, only the following types of productions will be allowed:

$$\begin{aligned} &1. \quad H \rightarrow G_1 G_2 \\ &2. \quad H \rightarrow V \end{aligned} \tag{1}$$

The grammar is stochastic, because to each rule there is assigned a probability of its use. Let H be any nonterminal, and let $\#(H)$ be the number of productions rewriting H. The ith of these productions will then take place with probability $P(i \mid H)$. It is assumed that for all $i = 1, 2, \ldots, \#(H)$, $P(i \mid H)$ is a strictly positive number and that

$$\sum_{i=1}^{\#(H)} P(i \mid H) = 1 \tag{2}$$

It will be convenient to denote the probabilities $P(i \mid H)$ by the productions they refer to, e.g., $P(H \rightarrow G_1 G_2)$ or $P(H \rightarrow V)$.

A context-free grammar is assumed to generate sentences from top to bottom, starting with some rule $s \rightarrow G_1 G_2$ that rewrites the sentence symbol s and is used with probability $P(s \rightarrow G_1 G_2)$. The generated nonterminals G_1 and G_2 are then rewritten, and the rewriting process continues until no nonterminals remain to be rewritten, all having been replaced by words through use of rewrite rules of type (1). The probability of the entire process is equal to the product of the probabilities of the individual rewrite rules used.

We say that a SCFG is well defined in case it forms a *language model*; that is, the total probability of strings of terminals generated by the grammar is equal to 1:

$$\sum_{n=1}^{\infty} \sum_{w_1 w_2 \ldots w_n \in \mathcal{V}^*} P(s \rightarrow w_1 w_2 \ldots w_n) = 1$$

A context-free grammar is said to be *proper* if starting from the distinguished nonterminal s, the only nonterminals produced are those whose further rewriting can eventually result in a string of terminals. In fact, condition (2) is necessary and sufficient for a SCFG to be well defined if the underlying grammar is proper.[2]

The solutions to the following four problems are of interest.

2 The following simple algorithm determines whether or not a grammar may be made proper by the elimination of rules.

Let $\mathcal{S}$ be the set of all nonterminals H such that a rule $H \rightarrow V$ exists for some nonterminal V.

1. What is the probability $P(s \rightarrow w_1 w_2 \ldots w_n)$ that the grammar, beginning with the start nonterminal s, generates a given word string (sentence) $w_1 w_2 \ldots w_n$, $w_i \in \mathcal{V}$?

The desired probability is computed by the Inside Algorithm (Baker 1979), which is a modification of the well-known CYK parsing algorithm (Younger 1967; Graham et al. 1980).

2. What is the most probable parse of a given word string $w_1 w_2 \ldots w_k$? That is, which sequence of rewrite rules resulting in $w_1 w_2 \ldots w_k$ is such that the product of its probabilities is maximal?

This parse is computed by the Viterbi Algorithm (Jelinek 1985), which uses the same chart as the CYK algorithm.

3. What is the probability $P(s \rightarrow w_1 w_2 \ldots w_n \ldots)$ that the grammar, beginning with the start nonterminal s, generates a word string (sentence) whose initial substring is $w_1 w_2 \ldots w_n$?

The algorithm providing the answer to this question is developed in the present paper.

4. Given the set of rules specifying a context-free grammar, how should the probabilities of their use be determined?

An answer to this question requires a criterion by which to judge it. The maximum likelihood criterion is as follows: given a "training corpus" $\mathcal{W}_T$ (that is, a set of sentences), determine the production probabilities so as to maximize the probability that the grammar generated $\mathcal{W}_T$. The Inside–Outside Algorithm (Baker 1979) extracts probabilities that locally (i.e., not necessarily globally) maximize the likelihood of $\mathcal{W}_T$.

3. Development of the Left-to-Right Inside (LRI) Algorithm

In this section we will develop the Left-to-Right (LRI) Algorithm, which will allow us to calculate the desired probabilities $P(s \rightarrow w_1 w_2 \ldots w_k \ldots)$. In order to present the LRI Algorithm, we will introduce some notation that will simplify the appearance of the following formulas. Let $P(H\langle i,j\rangle)$ denote the probability $P(H \rightarrow w_i \ldots w_j)$ that starting with the nonterminal H, successive application of grammar rules has produced the sequence $w_i w_{i+1} \ldots w_j$. That is, if the SCFG production process is represented by the usual tree diagram, then $P(H\langle i,j\rangle)$ is the sum of the probabilities of all trees whose root is H and whose leaves are $w_i, w_{i+1}, \ldots, w_j$.

---

1. If $\mathcal{S} = \mathcal{G}$, the grammar is proper. Stop.

   Else if $\mathcal{S} \neq \mathcal{G}$, then find the set $\mathcal{A}$ of all nonterminals H not belonging to $\mathcal{S}$ that rewrite as $H \rightarrow G_1 G_2$ with G_1 and G_2 belonging to $\mathcal{S}$.

2. If $\mathcal{A}$ is not empty, include the set $\mathcal{A}$ in $\mathcal{S}$ and go to 1.

3. If $s \in \mathcal{S}$, eliminate from $\mathcal{G}$ all nonterminals not belonging to $\mathcal{S}$ and purge all rules involving nonterminals not belonging to $\mathcal{S}$. The resulting grammar is proper.

   Else if $s \notin \mathcal{S}$, the grammar cannot be made proper by purging.

Next, let $P(H \ll i,j)$ denote the sum of the probabilities of all trees with root node H resulting in word strings whose initial substring is $w_i w_{i+1} \ldots w_j$. Thus

$$\begin{aligned} P(H \ll i,j) &= P(H\langle i,j\rangle) + \sum_{x_1} P(H \rightarrow w_i \ldots w_j x_1) \\ &\quad + \sum_{x_1 x_2} P(H \rightarrow w_i \ldots w_j x_1 x_2) + \cdots \\ &\quad + \sum_{x_1 \ldots x_n} P(H \rightarrow w_i \ldots w_j x_1 \ldots x_n) + \cdots \end{aligned} \tag{3}$$

Note that the first sum in (3) is over all possible words x_1, the second is over all possible word pairs $x_1 x_2$, and the third sum (the general term) is over all possible word n-tuples $x_1 x_2 \ldots x_n$. Using the notation (3), the desired probability $P(s \rightarrow w_1 w_2 \ldots w_k \ldots)$ is denoted by $P(s \ll 1,k)$.

In what follows we will need $P_L(H \rightarrow G)$, the sum of the probabilities of all the rules $H \rightarrow G_1 G_2$ whose first righthand side element is $G_1 = G$. That is,

$$P_L(H \rightarrow G) = \sum_{G_2} P(H \rightarrow G G_2) \tag{4}$$

Next we define the quantity

$$\begin{aligned} Q_L(H \Rightarrow G) &= P_L(H \rightarrow G) + \sum_{A_1} P_L(H \rightarrow A_1) P_L(A_1 \rightarrow G) \\ &\quad + \sum_{A_1,A_2} P_L(H \rightarrow A_1) P_L(A_1 \rightarrow A_2) P_L(A_2 \rightarrow G) + \cdots \\ &\quad + \sum_{A_1,\ldots,A_k} P_L(H \rightarrow A_1) P_L(A_1 \rightarrow A_2) \ldots P_L(A_k \rightarrow G) + \cdots \\ &= \sum_{\alpha} P(H \stackrel{*}{\Rightarrow} G\alpha) \end{aligned} \tag{5}$$

which is the sum of probabilities of all trees with root node H that produce G as the leftmost (first) nonterminal. Note that the last displayed (general) term accounts for all trees whose leftmost leaf has depth k. Note further that the above sum converges since we assume that our underlying grammar is proper, and that rule probabilities are non-zero.

We are now ready to compute

$$\begin{aligned} P(H \ll i,i) = P(H \rightarrow w_i) &+ \sum_{G} P_L(H \rightarrow G) P(G \rightarrow w_i) \\ &+ \sum_{G} \sum_{A_1} P_L(H \rightarrow A_1) P_L(A_1 \rightarrow G) P(G \rightarrow w_i) \\ &+ \sum_{G} \sum_{A_1,A_2} P_L(H \rightarrow A_1) P_L(A_1 \rightarrow A_2) P_L(A_2 \rightarrow G) P(G \rightarrow w_i) \\ &+ \ldots \\ &+ \sum_{G} \sum_{A_1,\ldots,A_k} P_L(H \rightarrow A_1) P_L(A_1 \rightarrow A_2) \ldots P_L(A_k \rightarrow G) P(G \rightarrow w_i) \\ &+ \ldots \end{aligned}$$

Thus, using definition (5) we get

$$P(H \ll i,i) = P(H \rightarrow w_i) + \sum_G Q_L(H \Rightarrow G)P(G \rightarrow w_i) \tag{6}$$

To compute $P(H \ll i, i+n)$ for $n > 0$, we will need to define

$$Q_L(H \Rightarrow G_1G_2) = P(H \rightarrow G_1G_2) + \sum_A Q_L(H \Rightarrow A)P(A \rightarrow G_1G_2) \tag{7}$$

which can be seen to be the sum of probabilities of all trees with root node H whose last leftmost production results in leaves G_1 and G_2. To compute $P(H \ll i, i+n)$ we will rely on the strict *CN* form of the grammar. Obviously,

$$\begin{aligned} P(H \ll i,i+n) \;=\; & \sum_{G_1,G_2} P(H \rightarrow G_1G_2)\,\big[P(G_1\langle i,i\rangle)P(G_2 \ll i+1, i+n) \\ & + P(G_1\langle i,i+1\rangle)P(G_2 \ll i+2, i+n) + \cdots \\ & + P(G_1\langle i,i+n-1\rangle)P(G_2 \ll i+n, i+n) \\ & + P(G_1 \ll i, i+n)\big] \end{aligned} \tag{8}$$

since to generate the initial substring $w_iw_{i+1}\ldots w_{i+n}$, some rule $H \rightarrow G_1G_2$ must first be applied and then the first part of the substring must be generated from G_1 and its remaining part (and perhaps more!) from G_2.

Defining the function

$$\begin{aligned} R(G_1,G_2) \;=\; & \big[P(G_1\langle i,i\rangle)P(G_2 \ll i+1, i+n) + P(G_1\langle i,i+1\rangle)P(G_2 \ll i+2, i+n) + \cdots \\ & + P(G_1\langle i,i+n-1\rangle)P(G_2 \ll i+n, i+n)\big] \end{aligned} \tag{9}$$

we can next rearrange (8) as follows:

$$\begin{aligned} P(H \ll i,i+n) \;=\; & \sum_{G_1,G_2} P(H \rightarrow G_1G_2)R(G_1,G_2) \\ & + \sum_{A_1} P_L(H \rightarrow A_1)P(A_1 \ll i, i+n) \end{aligned} \tag{10}$$

where we took advantage of the definition (4) and denoted the variable in the last sum by A_1 instead of by G_1.

Renaming H in (10) as A_1, and A_1 as A_2, we get

$$\begin{aligned} P(A_1 \ll i,i+n) \;=\; & \sum_{G_1,G_2} P(A_1 \rightarrow G_1G_2)R(G_1,G_2) \\ & + \sum_{A_2} P_L(A_1 \rightarrow A_2)P(A_2 \ll i, i+n) \end{aligned} \tag{11}$$

Substituting (11) into (10) and collecting and factoring out common terms, we get

$$P(H \ll i, i+n) = \sum_{G_1,G_2} \left[P(H \to G_1G_2) + \sum_{A_1} P_L(H \to A_1)P(A_1 \to G_1G_2) \right] R(G_1, G_2) + \sum_{A_1,A_2} P_L(H \to A_1)P_L(A_1 \to A_2)P(A_2 \ll i, i+n) \quad (12)$$

Next, renaming A_1 in (11) as A_2, and A_2 as A_3, and substituting the result into (12), we get

$$P(H \ll i, i+n) = \sum_{G_1,G_2} \left[P(H \to G_1G_2) + \sum_{A_1} P_L(H \to A_1)P(A_1 \to G_1G_2) + \sum_{A_1,A_2} P_L(H \to A_1)P_L(A_1 \to A_2)P(A_2 \to G_1G_2) \right] R(G_1, G_2) + \sum_{A_1,A_2,A_3} P_L(H \to A_1)P_L(A_1 \to A_2)P_L(A_2 \to A_3)P(A_3 \ll i, i+n) \quad (13)$$

The pattern is now clear. Since

$$\sum_{A_1,\ldots,A_k} P_L(H \to A_1)P_L(A_1 \to A_2)\ldots P_L(A_{k-1} \to A_k)P(A_k \ll i, i+n)$$

tends to 0 as k grows without limit, then using definition (7) and successive re-substitutions, we get the final formula

$$P(H \ll i, i+n) = \sum_{G_1,G_2} Q_L(H \Rightarrow G_1G_2)R(G_1G_2) \quad (14)$$

$$= \sum_{G_1,G_2} Q_L(H \Rightarrow G_1G_2) \left[\sum_{j=1}^{n} P(G_1\langle i, i+j-1\rangle)P(G_2 \ll i+j, i+n) \right]$$

where the last equality follows from (9), the definition of $R(G_1, G_2)$.

We can now notice that formula (14) is very similar to the well-known formula

$$P(H\langle i, i+n\rangle) = \sum_{G_1,G_2} P(H \to G_1G_2) \left[\sum_{j=1}^{n} P(G_1\langle i, i+j-1\rangle)P(G_2\langle i+j, i+n\rangle) \right] \quad (15)$$

that allows an iterative calculation of the (inside) probabilities $P(H\langle i, i+n\rangle)$ ((15) serves as the basis for the Inside Algorithm (Baker 1979)). There are two differences between (14) and (15): instead of the rule probability $P(H \to G_1G_2)$ in (15), we have in (14) the sum-of-tree-probability function $Q_L(H \Rightarrow G_1G_2)$ (defined in (7)), and instead of the simple span generation probability $P(G_2\langle i+j, i+n\rangle)$ in (15), we have in (14) the initial substring generation probability $P(G_2 \ll i+j, i+n)$ (defined in (3)). It follows that once we determine how to calculate the values of $Q_L(H \Rightarrow G_1G_2)$ (this is discussed in the next section), we will be able to compute iteratively all the other quantities (that

is, $P(H \ll i,j)$ and $P(H\langle i,j\rangle)$). In fact, it follows from (14) that to calculate $P(s \ll 1,k)$ one proceeds as follows:

1. Calculate probabilities $P(G\langle i,i+n\rangle)$ for $i = 1,2,\ldots,k-1$, $n = 0,1,2,\ldots,k-i-1$, iteratively by formula (15).
2. Calculate probabilities $P(H \ll k,k)$ by formula (6).
3. Calculate probabilities

$$P(H \ll k-1,k) = \sum_{G_1,G_2} Q_L(H \Rightarrow G_1G_2)P(G_1\langle k-1,k-1\rangle)P(G_2 \ll k,k)$$

4. Calculate probabilities

$$P(H \ll k-2,k) = \sum_{G_1,G_2} Q_L(H \Rightarrow G_1G_2) \left[\sum_{j=1}^{2} P(G_1\langle k-2,k+j-3\rangle)P(G_2 \ll k+j-2,k)\right]$$

$\vdots$

k. Calculate probabilities

$$P(H \ll 2,k) = \sum_{G_1,G_2} Q_L(H \Rightarrow G_1G_2) \left[\sum_{j=1}^{k-2} P(G_1\langle 2,1+j\rangle)P(G_2 \ll 2+j,k)\right]$$

k+1. Calculate the probability

$$P(s \ll 1,k) = \sum_{G_1,G_2} Q_L(s \rightarrow G_1G_2) \left[\sum_{j=1}^{k-1} P(G_1\langle 1,j\rangle)P(G_2 \ll j+1,k)\right]$$

4. Determination of the Functions $Q_L(H \Rightarrow G_1G_2)$ and $P(H \ll i,i)$

Let us first observe that if $w_i = v$ then $P(H \ll i,i) = P(H \rightarrow v\ldots)$ which, consistent with previous notation (5), we denote by $Q_L(H \Rightarrow v)$. We then get from (6)

$$Q_L(H \Rightarrow v) = P(H \rightarrow v) + \sum_{G} Q_L(H \Rightarrow G)P(G \rightarrow v) \tag{16}$$

It follows from (16) and (7) that to calculate the desired quantities $P(H \ll i,i)$ and $Q_L(H \Rightarrow G_1G_2)$ we must first determine the left corner probability sums $Q_L(H \Rightarrow G)$. We will use matrix algebra to compute them.

Let $\mathbf{P_L}$ and $\mathbf{Q_L}$ denote the square matrices (their dimension is equal to the number of nonterminals) whose elements in the Hth row and Gth column are $P_L(H \rightarrow G)$ (defined in (4)) and $Q_L(H \Rightarrow G)$, respectively. Then equation (5) can be rewritten in matrix form as

$$\mathbf{Q_L} = \mathbf{P_L} + \mathbf{P_L^2} + \mathbf{P_L^3} + \ldots \mathbf{P_L^k} + \ldots \tag{17}$$

where $\mathbf{P}_L^i$ denotes *i-fold* multiplication of the matrix $\mathbf{P}_L$ with itself. Post-multiplying both sides of (17) by the matrix $\mathbf{P}_L$, subtracting the resulting equation from (17), and cancelling terms, we get

$$\mathbf{Q}_L - \mathbf{Q}_L\mathbf{P}_L = \mathbf{P}_L \tag{18}$$

Finally, denoting by $\mathbf{I}$ the diagonal unit matrix of the same dimension as $\mathbf{P}_L$, we get from (18) the desired solution

$$\mathbf{Q}_L = \mathbf{P}_L[\mathbf{I} - \mathbf{P}_L]^{-1} \tag{19}$$

where $[\mathbf{I} - \mathbf{P}_L]^{-1}$ denotes the inverse of the matrix $[\mathbf{I} - \mathbf{P}_L]$.

Equation (16) can also be stated in matrix form. Denoting by P_W and Q_W the rectangular matrices with elements $P(H \rightarrow w)$ and $Q_L(H \rightarrow w)$ in the *Hth* row and *wth* column, respectively, we get from (16) that

$$\mathbf{Q}_W = [\mathbf{I} + \mathbf{Q}_L]\mathbf{P}_W \tag{20}$$

5. Conclusion

While the LRI algorithm together with formulas (19) and (20) constitutes the solution to the stated problem, its practicality is limited to grammars whose total number of nonterminals is sufficiently limited so as to allow the calculation of the inverse $[\mathbf{I} - \mathbf{P}_L]^{-1}$.

The algorithm itself has exactly twice the complexity of the Inside Algorithm computing $P(H\langle i, i+n\rangle)$ by formula (15), and is thus of order n^3. In fact, once all the probabilities required for the computation of $P(s \ll 1, k)$ are computed, to get the next probability of interest, $P(s \ll 1, k+1)$, one needs to compute the following quantities:

1. The probabilities $P(G\langle i, k\rangle)$ for $i = k, k-1, \ldots, 1$, in that order.
2. The probabilities $P(H \ll i, k+1)$ for $i = k+1, k, \ldots, 2$, in that order.
3. The probability $P(s \ll 1, k+1)$.

Let us finally recall that the language model of speech recognition provides to the recognizer the probability $P(W_k = v \mid w_1 w_2 \ldots w_{k-1})$ for all possible words v, and that we therefore must be able to compute the probability $P(s \rightarrow w_1 w_2 \ldots w_{k-1} v \ldots)$ for all N words v of the vocabulary. Fortunately, this does not mean carrying out the LRI algorithm N times for each word position k, but only M times, where M is the number of nonterminals of the grammar.

In fact, a simple modification of the algorithm allows one to compute the probabilities of $P(s \rightarrow w_1 w_2 \ldots w_{k-1} g_i \ldots)$ where g_i is an element of the set of nonterminals $\mathcal{G} = \{g_1 = s, g_2, \ldots, g_M\}$. This may be done, for example, by setting

$$P(H \ll k, k) = \begin{cases} 1 & \text{if } H = g_i \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

in the algorithm of Section 3. Our desired LRI probabilities can then be computed by the formula

$$P(s \rightarrow w_1 w_2 \ldots w_{k-1} v \ldots) = \sum_{i=1}^{M} Q_L(g_i \Rightarrow v) P(s \rightarrow w_1 w_2 \ldots w_{k-1} g_i \ldots) \tag{22}$$

This modification is particularly practical when the size of the vocabulary greatly exceeds the number of nonterminals in the grammar.

References

Bahl, L. R.; Jelinek, F.; and Mercer, R. L. (1983). "A maximum likelihood approach to continuous speech recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol PAMI-5, No 2, 1798–1790.

Baker, J. K. (1979). "Trainable grammars for speech recognition." *Proceedings, Spring Conference of the Acoustical Society of America*, Boston, MA, 547–550.

Graham, S. L.; Harrison, M. A.; and Ruzzo, W. L. (1980). "An improved context-free recognizer," *ACM Transactions on Programming Languages and Systems*, Vol 2, No 3, 415–462.

Jelinek, F. (1985). "Markov source modeling of text generation." In *The Impact of Processing Techniques on Communications*, edited by J. K. Skwirzinski. Dordrecht: Nijhoff.

Wright, J. H.; and Wrigley, E. N. (1989). "Probabilistic LR parsing for speech recognition." International Workshop on Parsing Technologies. 105–114.

Younger, D. H. (1967). "Recognition and parsing of context free languages in time N^3," *Information and Control* 10, 1980–208.